Alexander Garcia June 19, 2017

Assignment-4

1. Data: $[(0,0),(1,1),(2,3)]$

   (a) A full degree polynomial interpolation using the Lagrange basis is by definition

$$y = \sum_{i=1}^{n} y_i L_i(x_i)$$

where

$$L_i = \prod_{k=1 \;\; k \neq i}^{n} \frac{x - x_k}{x_i - x_k}$$

In this case, $n = 3$ and
$x_1 = 0 \quad x_2 = 1 \quad x_3 = 2$
$y_1 = 0 \quad y_2 = 1 \quad y_3 = 3$

$y = 0L_1(x_1) + 1L_2(x_2) + 3L_3(x_3)$

$$L_2 = \frac{x-0}{1-0} * \frac{x-2}{1-2} = -x^2 + 2x$$

$$L_3 = \frac{x-0}{2-0} * \frac{x-1}{2-1} = \frac{x^2 - x}{2}$$

$$y = -x^2 + 2x + \frac{3}{2}(x^2 - x) = \frac{1}{2}(x^2 + x)$$

Just to ensure the interpolant agrees with the data, a small MATLAB script was used to plot y with the points overlaid.

`holmes5_1.m` script

```
% check for Holmes 5.1

% define calculated interpolant
y = (1/2)*(x.^2 + x);
x = linspace(0,3);

% display results, including original data points
plot(x,y,0,0,'*',1,1,'*',2,3,'*');
legend('Interpolant','(x_1,y_1)','(x_2,y_2)','(x_3,y_3)','Location','Northwest');
```

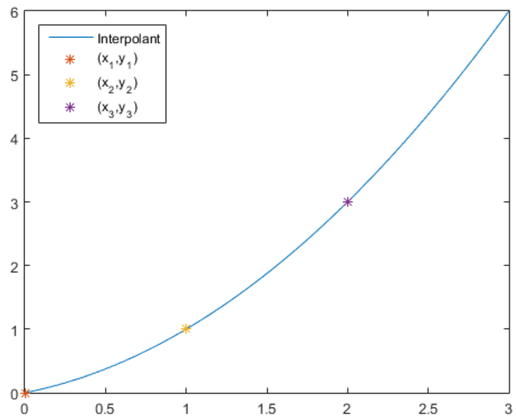

Figure 1: Graph of y and the data points

(b) Piecewise linear interpolation for this data set consists of 2 equations $S_1(x), S_2(x)$ which interpolate the data between $[x_1, x_2]$, $[x_2, x_3]$ respectively.

In general,

$$S_i(x) = a_i x + b_i$$

where

$$S_i(x_i) = y_i$$

From here, we get

$$a_i = \frac{y_{i+1} - y_i}{x_{i+1} - x_i} \qquad b_i = y_i - \frac{y_{i+1} - y_i}{x_{i+1} - x_i} * x_i$$

Then,

$$S_1(x) = \frac{1-0}{1-0}x + (0 - \frac{1-0}{1-0} * 0) = x \qquad x \in [0, 1]$$

and

$$S_2(x) = \frac{3-1}{2-1}x + (1 - \frac{3-1}{2-1} * 1) = 2x - 1 \qquad x \in [1, 2]$$

A simple script `holmes5_1_lin.m` was used to check the results of the interpolation.

`holmes5_1_lin.m`

```
% script to check part (b) of Holmes 5.1

% define domain
x1 = linspace(0,1);
x2 = linspace(1,2);

% define functions
S1 = x1;
S2 = 2*x2 - 1;
```

```
% display results, including original data points
plot(x1,S1,x2,S2,0,0,'*',1,1,'*',2,3,'*');
legend('S1','S2','(x_1,y_1)','(x_2,y_2)','(x_3,y_3)','Location','Northwest');
```

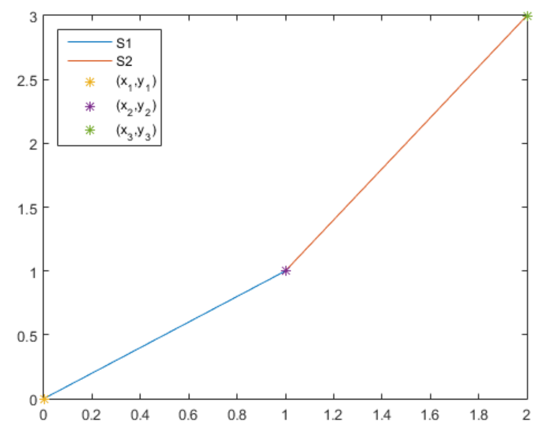

Figure 2: Graph of linear piecewise interpolant

(c) Natural cubic spline interpolation insists that the second derivative at both the first and last data points are zero.

Again, there are 2 equations $S_1(x), S_2(x)$ that interpolate the data between $[x_1, x_2]$, $[x_2, x_3]$ respectively. In general,

$$S_i(x) = a_i x^3 + b_i x^2 + c_i x + d_i$$

In order to satisfy the interpolation conditions, it must be enforced that $S_i(x_i) = y_i$. In addition, to ensure a smooth curve through the data, we insist that $S_i(x_i) = S_{i+1}(x_i)$, $S_i'(x_i) = S_{i+1}'(x_i)$, $S_i''(x_i) = S_{i+1}''(x_i)$

Given the initial conditions, we can construct a matrix to solve for the coefficients a_i, b_i, c_i, d_i. Each entry in matrix $\mathbf{A}$ is the value of x_i raised to the corresponding power.

$$\begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 8 & 4 & 2 & 1 \\ 6 & 2 & 0 & 0 & -6 & -2 & 0 & 0 \\ 3 & 2 & 1 & 0 & -3 & -2 & -1 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 12 & 4 & 0 & 0 \end{bmatrix} \begin{bmatrix} a_1 \\ b_1 \\ c_1 \\ d_1 \\ a_2 \\ b_2 \\ c_2 \\ d_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 3 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

A brief explanation of the contents of this matrix:

Rows 1-4:	Interpolation conditions; $S_i(x_i) = y_i$
Rows 5-6:	"Smoothness" conditions; agreeing first and second derivatives
Rows 7-8:	Natural spline conditions; zero end point second derivatives

Solving this system using the backslash command in MATLAB yields

$$\mathbf{c} = \begin{bmatrix}0 & 0 & 1 & 0 & 1 & -3 & 4 & -1\end{bmatrix}^T$$

Using these results, we have

$$S_1(x) = x$$

and

$$S_2(x) = x^3 - 3x^2 + 4x - 1$$

Below is the script used to both calculate and plot the resulting splines.

`holmes5_1_cubic.m`

```
% Script to check part (c) of Holmes5.1

% define coefficient matrix A
A = [ 0 0 0 1 0 0 0 0;
      1 1 1 1 0 0 0 0;
      0 0 0 0 1 1 1 1;
      0 0 0 0 8 4 2 1;
      6 2 0 0 -6 -2 0 0;
      3 2 1 0 -3 -2 -1 0;
      0 2 0 0 0 0 0 0;
      0 0 0 0 12 4 0 0; ];

% define resultant vector b
b = [0 1 1 3 0 0 0 0]';

% calculate vector of coefficients
c = A\b;

% define domains
x1 = linspace(0,1);
x2 = linspace(1,2);

% define splines from c
S1 = c(1)*x1.^3 + c(2)*x1.^2 + c(3)*x1 + c(4);
S2 = c(5)*x2.^3 + c(6)*x2.^2 + c(7)*x2 + c(8);

% display results, including original data points
plot(x1,S1,x2,S2,0,0,'*',1,1,'*',2,3,'*');
legend('S1','S2','(x_1,y_1)','(x_2,y_2)','(x_3,y_3)','Location','Northwest');
```

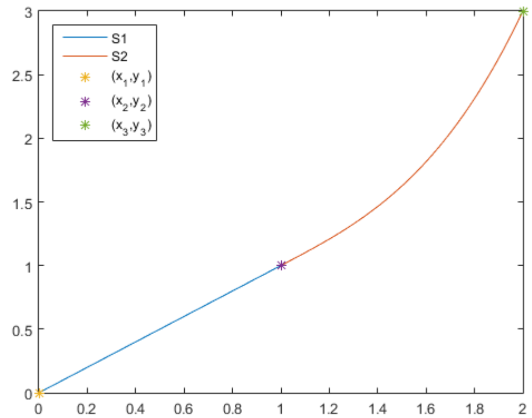

Figure 3: Graph of cubic piecewise interpolant

2. Function to interpolate: $f(x) = sinx$

   (a) Piecewise linear interpolation

   The linear spline passing through $\frac{\pi}{8}$ would be $S_1(x)$.

$$S_1(x) = \frac{y_2 - y_1}{x_2 - x_1}x + (y_1 - \frac{y_2 - y_1}{x_2 - x_1})x_1 \qquad x \in [0, \frac{\pi}{4}]$$

$$S_1(x) = \frac{\sqrt{2}/2 - 0}{\pi/4 - 0}x + (\sqrt{2}/2 - \frac{\sqrt{2}/2 - 0}{\pi/4 - 0})0$$
$$= \frac{\sqrt{2}/2}{\pi/4}x$$

   A MATLAB script `holmes5_5a.m` was used to calculate the estimated value of $f(\pi/8)$, and plot the interpolant and the original function.

$$S_1(\pi/8) = 0.35355$$
$$|S_1(\pi/8) - sin(\pi/8)| = 0.02913$$

`holmes5_5a.m`

```
% check linear interpolation of f(x) = sinx
clc;
% function definition
f = sin(x);
```

```
% calculated interpolant definition
s1 = ((sqrt(2)/2)/(pi/4))*x;

% display original plot, interpolant, and data points
x = linspace(0,pi/4);
plot(x,f,x,s1,0,0,'*',pi/4,sqrt(2)/2,'*');
legend('f(x)','S_1(x)','(x_1,y_1)','(x_2,y_2)','Location','Northwest');

% calculate error in interpolant at x = \pi/8
est = ((sqrt(2)/2)/(pi/4))*(pi/8);

err = est - sin(pi/8);
ea = abs(err);
fprintf('Estimated value of f(pi/8): %.5f\n', est);
fprintf('Absolute error in evaluating S_1(pi/8): %.5f\n', ea);
```

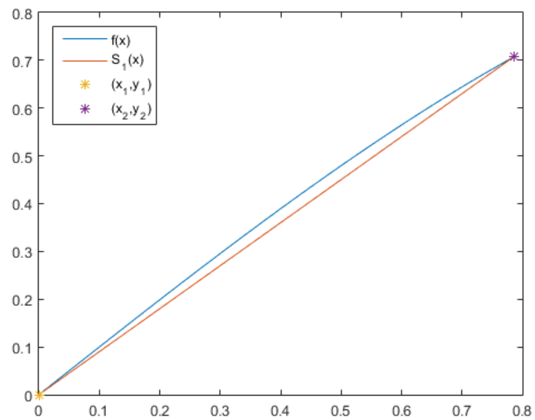

Figure 4: Graph of $f(x)$ and $S_1(x)$

(b) Full degree interpolation using the Lagrange basis

The Lagrange interpolant in this case uses $n = 3$, with data points

$x_1 = 0 \quad x_2 = \frac{\pi}{4} \quad x_3 = \frac{\pi}{2}$

$y_1 = 0 \quad y_2 = \frac{\sqrt{2}}{2} \quad y_3 = 1$

$y = 0L_1(x_1) + (\sqrt{2}/2)L_2(x_2) + 1L_3(x_3)$

$$L_2(x_2) = \frac{x-0}{\pi/4-0} * \frac{x-\pi/2}{\pi/4-\pi/2} = -\frac{16x^2-8\pi x}{\pi^2}$$

$$L_3(x_3) = \frac{x-0}{\pi/2-0} * \frac{x-\pi/4}{\pi/2-\pi/4} = \frac{8x^2-2\pi x}{\pi^2}$$

$$y = -\frac{\sqrt{2}}{2} * \frac{16x^2 - 8\pi x}{\pi^2} + \frac{8x^2 - 2\pi x}{\pi^2} = \frac{(8 - 8\sqrt{2})x^2 + (4\sqrt{2}\pi - 2\pi)x}{\pi^2}$$

Again, MATLAB was used to verify the result, as well as calculate the error.

$$y(\pi/8) = 0.40533$$

$$|y(\pi/8) - sin(\pi/8)| = 0.02265$$

holmes5_5b.m

```
% check lagrange interpolation of f(x) = sinx
clc;
% function definition
f = sin(x);

% calculated interpolant definition
s1 = ((8-8*sqrt(2))*x.^2+(4*sqrt(2)*pi-2*pi)*x)/pi^2;

% display original plot, interpolant, and data points
x = linspace(0,pi/2);
plot(x,f,x,s1,0,0,'*',pi/4,sqrt(2)/2,'*',pi/2,1,'*');
legend('f(x)','y','(x_1,y_1)','(x_2,y_2)','(x_3,y_3)','Location','Northwest');

% calculate error in interpolant at x = \pi/8
est = ((8-8*sqrt(2))*(pi/8)^2+(4*sqrt(2)*pi-2*pi)*(pi/8))/pi^2;

err = est - sin(pi/8);
ea = abs(err);
fprintf('Estimated value of f(pi/8): %.5f\n', est);
fprintf('Absolute error in evaluating y(pi/8): %.5f\n', ea);
```

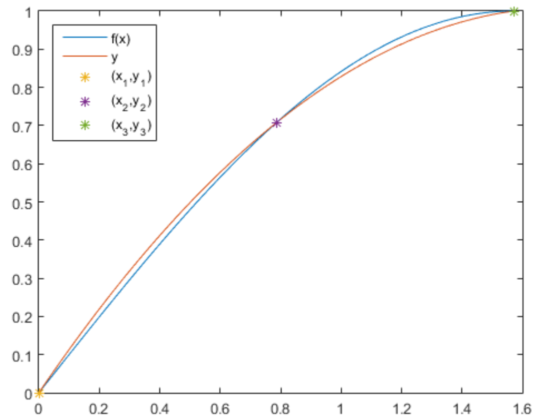

Figure 5: Graph of $f(x)$ and full degree interpolating polynomial y

(c) Natural cubic spline interpolation

Rows 1-4:	Interpolation conditions; $S_i(x_i) = y_i$
Rows 5-6:	"Smoothness" conditions; agreeing first and second derivatives
Rows 7-8:	Natural spline conditions; zero end point second derivatives

$$\begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ \pi^3/64 & \pi^2/16 & \pi/4 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \pi^3/64 & \pi^2/16 & \pi/4 & 1 \\ 0 & 0 & 0 & 0 & \pi^3/8 & \pi^2/4 & \pi/2 & 1 \\ 3\pi^2/16 & \pi/2 & 1 & 0 & -3\pi^2/16 & -\pi/2 & -1 & 0 \\ 3\pi/2 & 2 & 0 & 0 & -3\pi/2 & -2 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3\pi & 2 & 0 & 0 \end{bmatrix} \begin{bmatrix} a_1 \\ b_1 \\ c_1 \\ d_1 \\ a_2 \\ b_2 \\ c_2 \\ d_2 \end{bmatrix} = \begin{bmatrix} 0 \\ \sqrt{2}/2 \\ \sqrt{2}/2 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

A MATLAB script was used to calculate the resulting coefficients, the error of the interpolating spline, as well as plot the spline and $f(x)$. The resulting splines are

$$\begin{aligned} S_1(x) &= -0.21374x^3 + 1.03216x & x \in [0, \pi/4] \\ S_2(x) &= 0.21374x^3 - 1.00725x^2 + 1.82325x - 0.20711 & x \in [\pi/4, \pi/2] \end{aligned}$$

$$S_1(\pi/8) = 0.39239$$

$$|S_1(\pi/8) - sin(\pi/8)| = 0.00970$$

`holmes5_5c.m` script

```
% Script to check part (c) of Holmes5.5
clc;

% function definition
f = sin(x);

% define coefficient matrix A
A = [ 0 0 0 1 0 0 0 0;
      pi^3/64 pi^2/16 pi/4 1 0 0 0 0;
      0 0 0 0 pi^3/64 pi^2/16 pi/4 1;
      0 0 0 0 pi^3/8 pi^2/4 pi/2 1;
      3*pi^2/16 pi/2 1 0 -3*pi^2/16 -pi/2 -1 0;
      3*pi/2 2 0 0 -3*pi/2 -2 0 0;
      0 2 0 0 0 0 0 0;
      0 0 0 0 3*pi 2 0 0; ];

% define resultant vector b
b = [0 sqrt(2)/2 sqrt(2)/2 1 0 0 0 0]';

% calculate vector of coefficients
c = A\b;

fprintf('S1 = %.5fx^3 + %.5fx^2 + %.5fx + %.5f\n',c(1),c(2),c(3),c(4));
fprintf('S2 = %.5fx^3 + %.5fx^2 + %.5fx + %.5f\n',c(5),c(6),c(7),c(8));

% define domains
x = linspace(0,pi/2);
x1 = linspace(0,pi/4);
x2 = linspace(pi/4,pi/2);

% define splines from c
S1 = c(1)*x1.^3 + c(2)*x1.^2 + c(3)*x1 + c(4);
S2 = c(5)*x2.^3 + c(6)*x2.^2 + c(7)*x2 + c(8);

% display results, including original data points
plot(x,f,x1,S1,x2,S2,0,0,'*',pi/4,sqrt(2)/2,'*',pi/2,1,'*');
legend('f','S1','S2','(x_1,y_1)','(x_2,y_2)','(x_3,y_3)','Location','Northwest');

% calculate error in interpolant at x = \pi/8
est = c(1)*(pi/8)^3 + c(2)*(pi/8)^2 + c(3)*(pi/8) + c(4);

err = est - sin(pi/8);
ea = abs(err);
fprintf('Estimated value of f(pi/8): %.5f\n', est);
fprintf('Absolute error in evaluating S_1(pi/8): %.5f\n', ea);
```

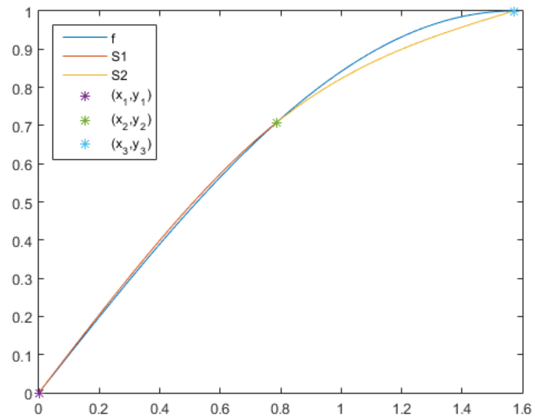

Figure 6: Graph of $f(x)$ and natural cubic splines $S_1(x), S_2(x)$

(d) Clamped cubic spline interpolation

Rows 1-4:	Interpolation conditions; $S_i(x_i) = y_i$
Rows 5-6:	"Smoothness" conditions; agreeing first and second derivatives
Rows 7-8:	Clamped spline conditions; fixed end point first derivatives

$$\begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ \pi^3/64 & \pi^2/16 & \pi/4 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \pi^3/64 & \pi^2/16 & \pi/4 & 1 \\ 0 & 0 & 0 & 0 & \pi^3/8 & \pi^2/4 & \pi/2 & 1 \\ 3\pi^2/16 & \pi/2 & 1 & 0 & -3\pi^2/16 & -\pi/2 & -1 & 0 \\ 3\pi/2 & 2 & 0 & 0 & -3\pi/2 & -2 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3\pi^2/4 & \pi & 1 & 0 \end{bmatrix} \begin{bmatrix} a_1 \\ b_1 \\ c_1 \\ d_1 \\ a_2 \\ b_2 \\ c_2 \\ d_2 \end{bmatrix} = \begin{bmatrix} 0 \\ \sqrt{2}/2 \\ \sqrt{2}/2 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Once again, the coefficients were calculated by a MATLAB script, as well as the error, and a plot of the spline and function.

$$\begin{aligned} S_1(x) &= -0.34178x^3 + 0.14151x^2 + x & x \in [0, \pi/4] \\ S_2(x) &= 0.49355x^3 - 1.82668x^2 + 2.54582x - 0.40469 & x \in [\pi/4, \pi/2] \end{aligned}$$

$$S_1(\pi/8) = 0.39382$$

$$|S_1(\pi/8) - sin(\pi/8)| = 0.01114$$

`holmes5_5d.m` script

```
% Script to check part (d) of Holmes5.5
clc;

% function definition
f = sin(x);

% define coefficient matrix A
A = [ 0 0 0 1 0 0 0 0;
     pi^3/64 pi^2/16 pi/4 1 0 0 0 0;
     0 0 0 0 pi^3/64 pi^2/16 pi/4 1;
     0 0 0 0 pi^3/8 pi^2/4 pi/2 1;
     3*pi^2/16 pi/2 1 0 -3*pi^2/16 -pi/2 -1 0;
     3*pi/2 2 0 0 -3*pi/2 -2 0 0;
     0 0 1 0 0 0 0 0;
     0 0 0 0 3*pi^2/4 2 0 0; ];

% define resultant vector b
b = [0 sqrt(2)/2 sqrt(2)/2 1 0 0 1 0]';

% calculate vector of coefficients
c = A\b;

fprintf('S1 = %.5fx^3 + %.5fx^2 + %.5fx + %.5f\n',c(1),c(2),c(3),c(4));
fprintf('S2 = %.5fx^3 + %.5fx^2 + %.5fx + %.5f\n',c(5),c(6),c(7),c(8));

% define domains
x = linspace(0,pi/2);
x1 = linspace(0,pi/4);
x2 = linspace(pi/4,pi/2);

% define splines from c
S1 = c(1)*x1.^3 + c(2)*x1.^2 + c(3)*x1 + c(4);
S2 = c(5)*x2.^3 + c(6)*x2.^2 + c(7)*x2 + c(8);

% display results, including original data points
plot(x,f,x1,S1,x2,S2,0,0,'*',pi/4,sqrt(2)/2,'*',pi/2,1,'*');
legend('f','S1','S2','(x_1,y_1)','(x_2,y_2)','(x_3,y_3)','Location','Northwest');

% calculate error in interpolant at x = \pi/8
est = c(1)*(pi/8)^3 + c(2)*(pi/8)^2 + c(3)*(pi/8) + c(4);

err = est - sin(pi/8);
ea = abs(err);
fprintf('Estimated value of f(pi/8): %.5f\n', est);
fprintf('Absolute error in evaluating S_1(pi/8): %.5f\n', ea);
```

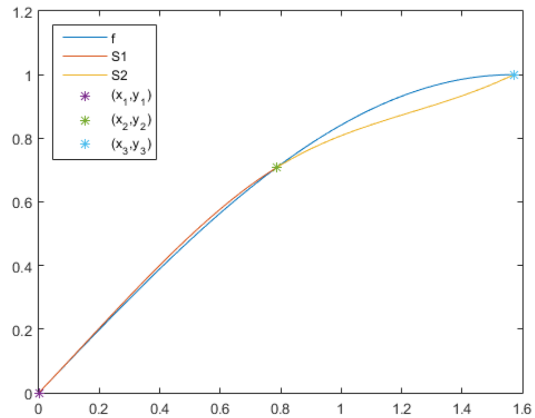

Figure 7: Graph of $f(x)$ and clamped cubic splines $S_1(x), S_2(x)$

(e) Chebyshev interpolation

The function needed to determine the nodes for full-degree interpolation are given by the zeros of the Chebyshev polynomial $T_n(x) = cosn\theta$, which occur at $n\theta = \frac{(2k-1)\pi}{2n}$, where $k = 1 : n$. However, this only applies to the interval $[-1, 1]$, and therefore must be expanded to our interval $[0, \pi/2]$. When generalized, the zeros lie at $\frac{b+a}{2} - \frac{b-a}{2} cos(\frac{(2k-1)\pi}{2n})$, again with $k = 1 : n$. In our case then, $n = 3$, $b = \pi/2$, $a = 0$, and the zeros are

$$\begin{aligned}
k = 1: &\quad x_1 = \tfrac{\pi}{4} - \tfrac{\pi}{4} cos(\tfrac{\pi}{6}) = 0.1052 \\
k = 2: &\quad x_2 = \tfrac{\pi}{4} - \tfrac{\pi}{4} cos(\tfrac{\pi}{2}) = \tfrac{\pi}{4} \\
k = 3: &\quad x_3 = \tfrac{\pi}{4} - \tfrac{\pi}{4} cos(\tfrac{5\pi}{6}) = 1.4656
\end{aligned}$$

We then use these nodes, to generate a full degree polynomial interpolant, in this case using the Lagrange basis.

$$y = y_1 L_1(x_1) + y_2 L_2(x_2) + y_3 L_3(x_3)$$

$$L_1(x_1) = \frac{x - \pi/4}{0.1052 - \pi/4} * \frac{x - 1.4656}{0.1052 - 1.4656}$$

$$L_2(x_2) = \frac{x - 0.1052}{\pi/4 - 0.1052} * \frac{x - 1.4656}{\pi/4 - 1.4656}$$

$$L_3(x_3) = \frac{x - 0.1052}{1.4656 - 0.1052} * \frac{x - \pi/4}{1.4656 - \pi/4}$$

$$y(x) = sin(x_1)L_1(x_1) + sin(x_2)L_2(x_2) + sin(x_3)L_3(x_3)$$

The function estimate, as well as the error, were calculated through a MATLAB script. The script also plots the interpolant and the original function.

$$y(\pi/8) = 0.39790$$

$$|y(\pi/8) - sin(\pi/8)| = 0.01521$$

`holmes5_5e.m` script

```
% script to calculate full degree interpolant based on
% ideal node spacing
clc;

% define domain
xx = linspace(0,pi/2);

% define original function
f = sin(xx);

% calculate ideal nodes
x1 = (pi/4)-(pi/4)*cos(pi/6);
x2 = (pi/4)-(pi/4)*cos(pi/2);
x3 = (pi/4)-(pi/4)*cos(5*pi/6);

% calculate lagrange basis polynomial based on ideal nodes
L1 = (xx-x2)./(x1-x2) .* (xx-x3)./(x1-x3);
L2 = (xx-x1)./(x2-x1) .* (xx-x3)./(x2-x3);
L3 = (xx-x1)./(x3-x1) .* (xx-x2)./(x3-x2);

y = sin(x1).*L1 + sin(x2).*L2 + sin(x3).*L3;

% display interpolant, original function, and data points
plot(xx,y,xx,f,x1,sin(x1),'*',x2,sin(x2),'*',x3,sin(x3),'*');
legend('y','f','(x_1,y_1)','(x_2,y_2)','(x_3,y_3)','Location','Northwest');

% calculate error in interpolant at y(\pi/8)
xx = pi/8;

L1 = (xx-x2)./(x1-x2) .* (xx-x3)./(x1-x3);
L2 = (xx-x1)./(x2-x1) .* (xx-x3)./(x2-x3);
L3 = (xx-x1)./(x3-x1) .* (xx-x2)./(x3-x2);

est = sin(x1).*L1 + sin(x2).*L2 + sin(x3).*L3;

err = est-sin(pi/8);
ea = abs(err);

fprintf('Estimated value of f(pi/8): %.5f\n',est);
fprintf('Absolute error in evaluating y(pi/8): %.5f\n',ea);
```

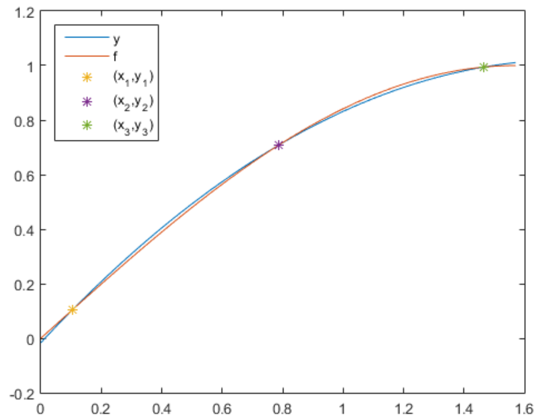

Figure 8: Graph of $f(x)$ and the Chebyshev polynomial $y(x)$

3.